Mississippi Computational Web Portal

Web Portal for
Distributed Marine Environment Forecast System
(DMEFS)

Tomasz Haupt

Mississippi State

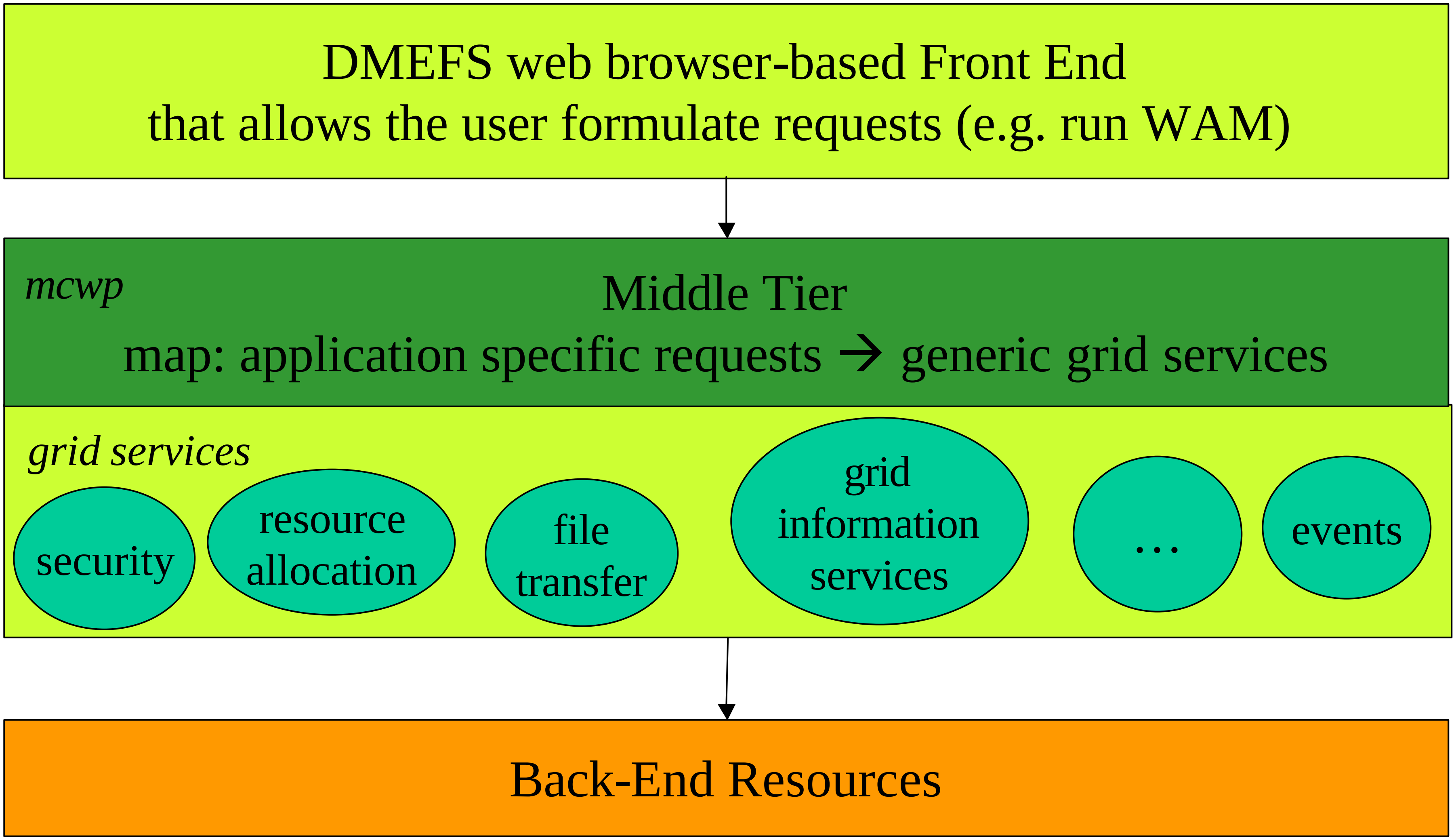
PURPOSE

1. **Secure seamless Web access** to remote computational resources
2. Intuitive front-end that **hides complexity** of heterogeneous, distributed backend systems.
3. Persistent, hierarchical, object oriented representation of computational tasks to enable **archiving, reusing, sharing, and transition for operational use.**
4. Support for **incorporating legacy applications.**
5. Integration of commodity software components (such as visualizations) into a **single, user friendly system.**
6. Support for model coupling.
7. Access to **remote databases.**

Grid Environment

- The Grid environment is very complex:
  - security (authentication, authorization, delegation of credentials,…)
  - (super) scheduling, (co)allocation
  - quality of service, fault tolerance
  - data transfer
  - network protocols
  - grid programming models,
  - …
- The end user often overwhelmed (if not intimidated)
- Very little grid applications

Portal Architecture



DMEFS web browser-based Front End
that allows the user formulate requests (e.g. run WAM)

mcwp

Middle Tier
map: application specific requests → generic grid services

grid services

security

resource allocation

file transfer

grid information services

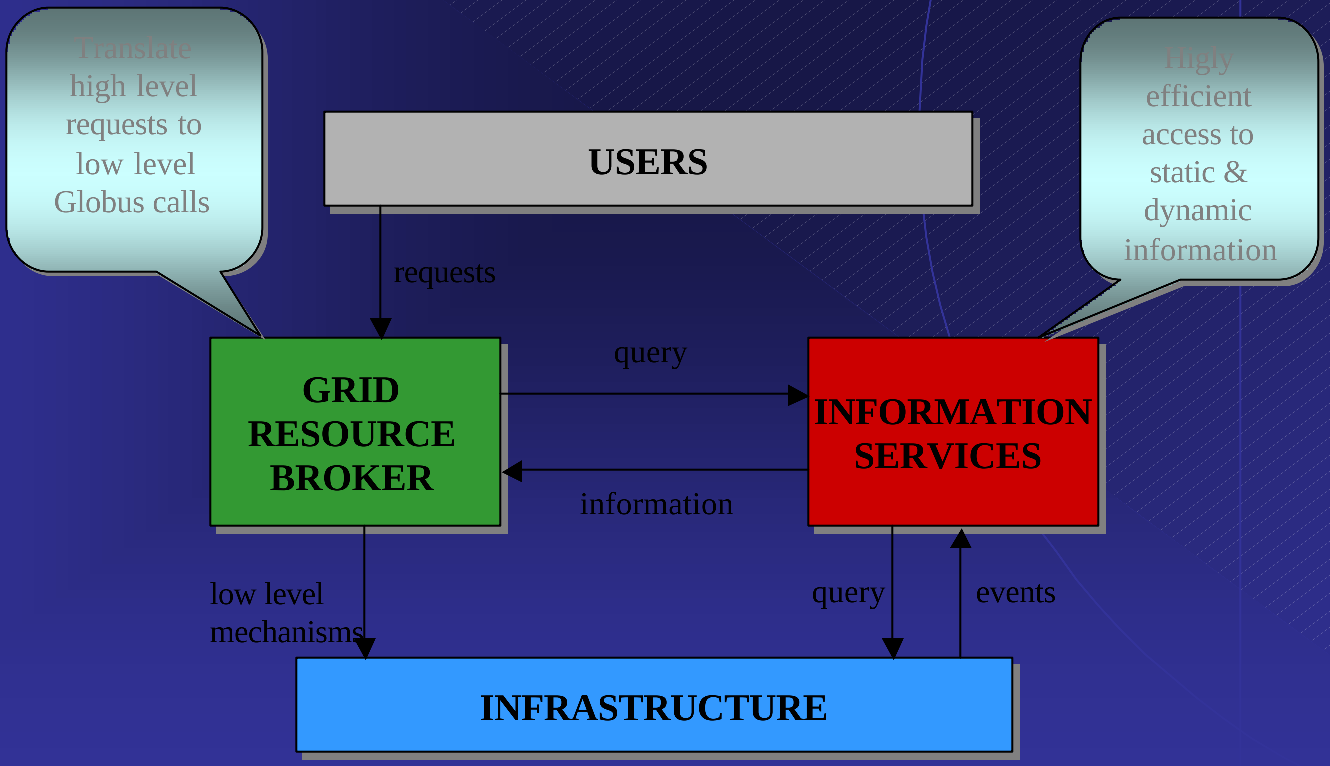
…

events

Back-End Resources

Grid Resource Broker

A portal to computational grids



Translate high level requests to low level Globus calls

USERS

requests

GRID RESOURCE BROKER

query

information

INFORMATION SERVICES

Higly efficient access to static & dynamic information

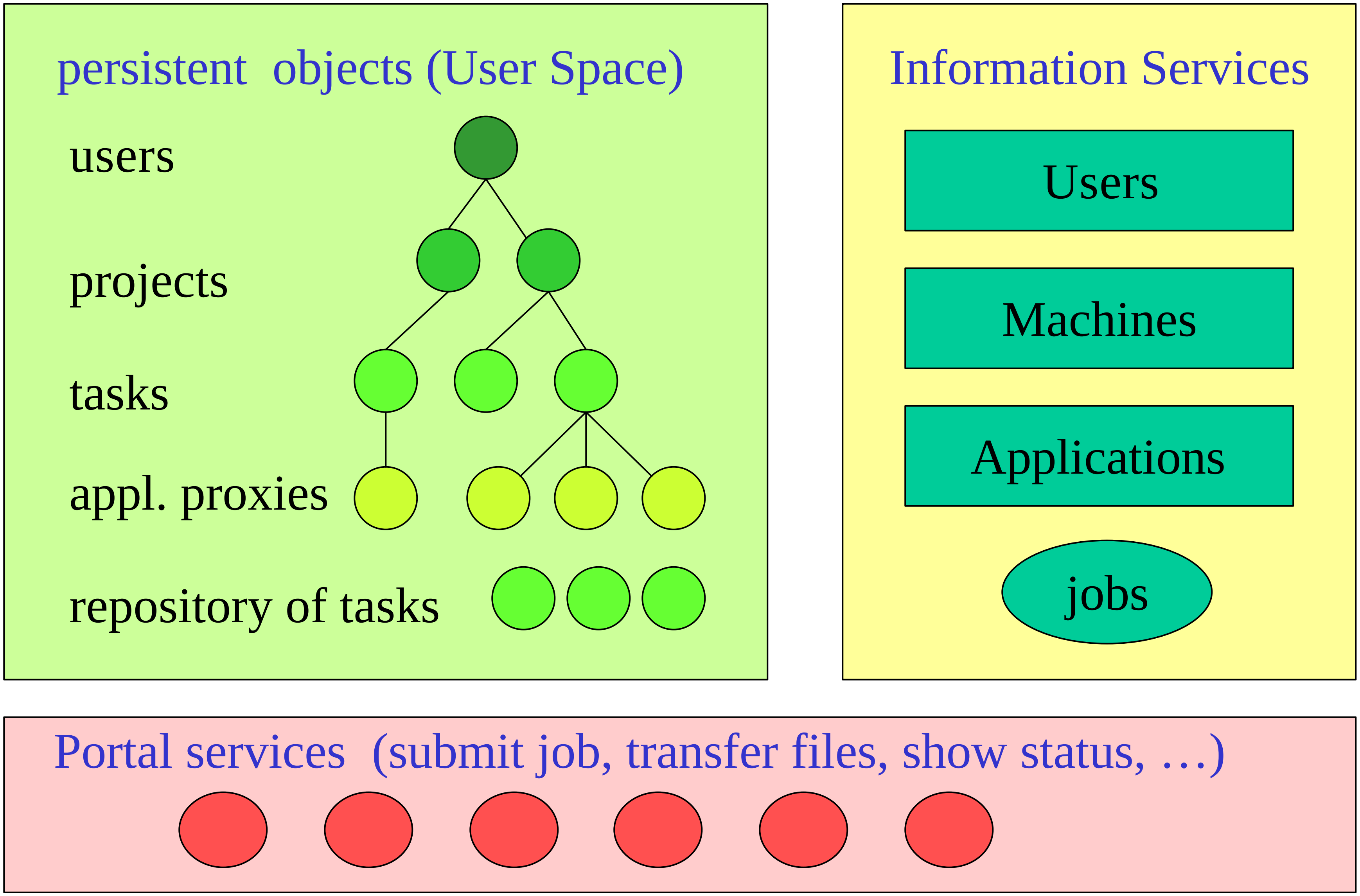
low level mechanisms

query

events

INFRASTRUCTURE

Elements of DMEFS Computational Portal



persistent objects (User Space)

users

projects

tasks

appl. proxies

repository of tasks

Information Services

Users

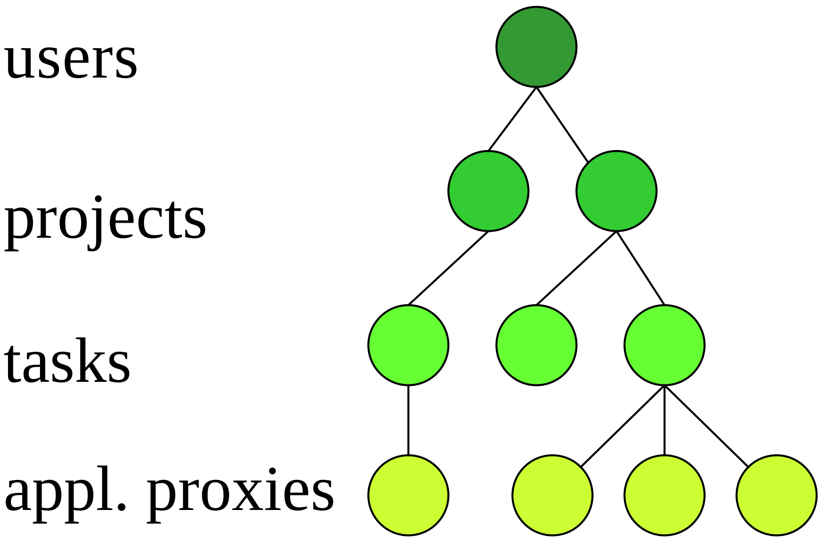
Machines

Applications

jobs

Portal services (submit job, transfer files, show status, …)

User Space

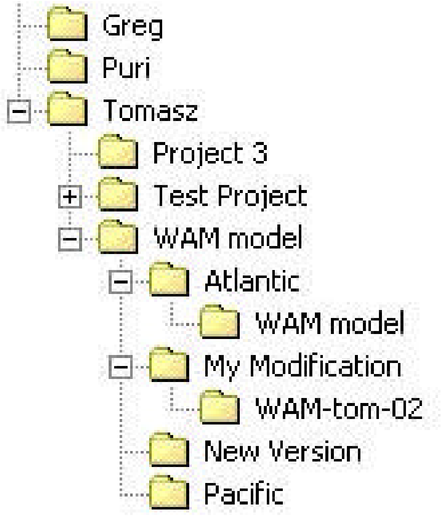


users

projects

tasks

appl. proxies



- Greg
- Puri
- Tomasz
  - Project 3
  - Test Project
  - WAM model
    - Atlantic
      - WAM model
    - My Modification
      - WAM-tom-02
    - New Version
    - Pacific

Organized as a tree – similar to a 4-level directory

User object: identity, authorization, preferences
Project object: a way to organize computational tasks
Task object: contains all information to submit a job
Application object: task constituent

EJB Components

- Entity Beans are persistent
- Session Beans operate on data

- *A textbook example:*
  - *A bank account is represented as an entity bean*
  - *To transfer funds between accounts, a session bean is used*
- *Note: EJB container enforces security, transaction processing as well as provides methods to find/select entity beans (accounts).*

Elements of DMEFS Computational Portal

persistent objects (User Space)

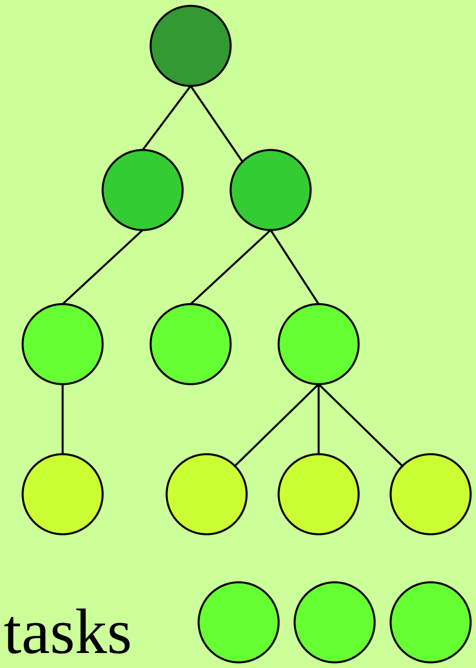
users

projects

tasks

appl. proxies

repository of tasks



Information Services

Users

Machines

Applications

jobs

Portal services (submit job, transfer files, show status, …)

Application descriptors

- name, description, keywords (queries, display)
- where it comes from (author, contact, …)
- what does it produce
  - for archiving
  - for visualizations
  - for model coupling (starting from dataflow)
- what it does need to run
  - input data
  - parameters (that GUI can be generated automatically, or preregistered GUI)
- where it is installed: all information to generate a batch script
- where can it be installed: all information to generate makefile
- events it generates

Task Descriptor

- hierarchical (from atomic to complex)
- contains “ ‘configured’ links” to application descriptors
- defines relationships between “atomic tasks”
- defines couples

How does it work: creating a task

login

create or select project

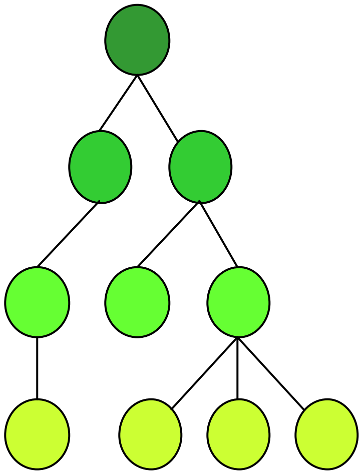
create or select task

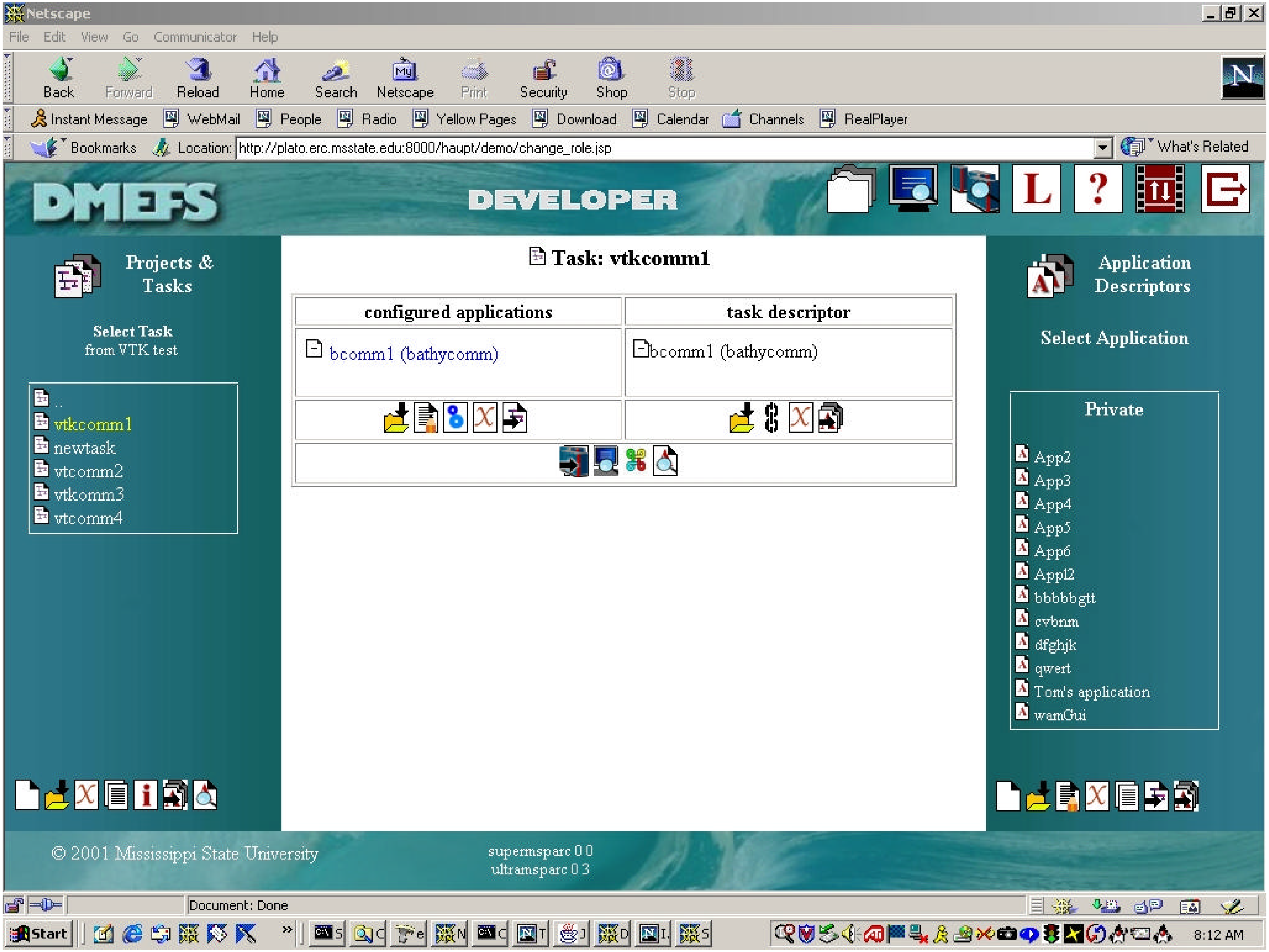
add one or more applications to the task

configure each application

link applications

submit the task




Netscape
File Edit View Go Communicator Help
Back Forward Reload Home Search Netscape Print Security Shop Stop
Instant Message WebMail People Radio Yellow Pages Download Calendar Channels RealPlayer
Bookmarks
What's Related
DMEFS
DEVELOPER
Document: Done
Start
8:12 AM

Location: http://plato.erc.msstate.edu:8000/haupt/demo/change_role.jsp

Projects & Tasks

Select Task
from VTK test

- ..
- vtkcomm1
- newtask
- vtcomm2
- vtkomm3
- vtcomm4

Task: vtkcomm1

configured applications	task descriptor
bcomm1 (bathycomm)	bcomm1 (bathycomm)

Application Descriptors

Select Application

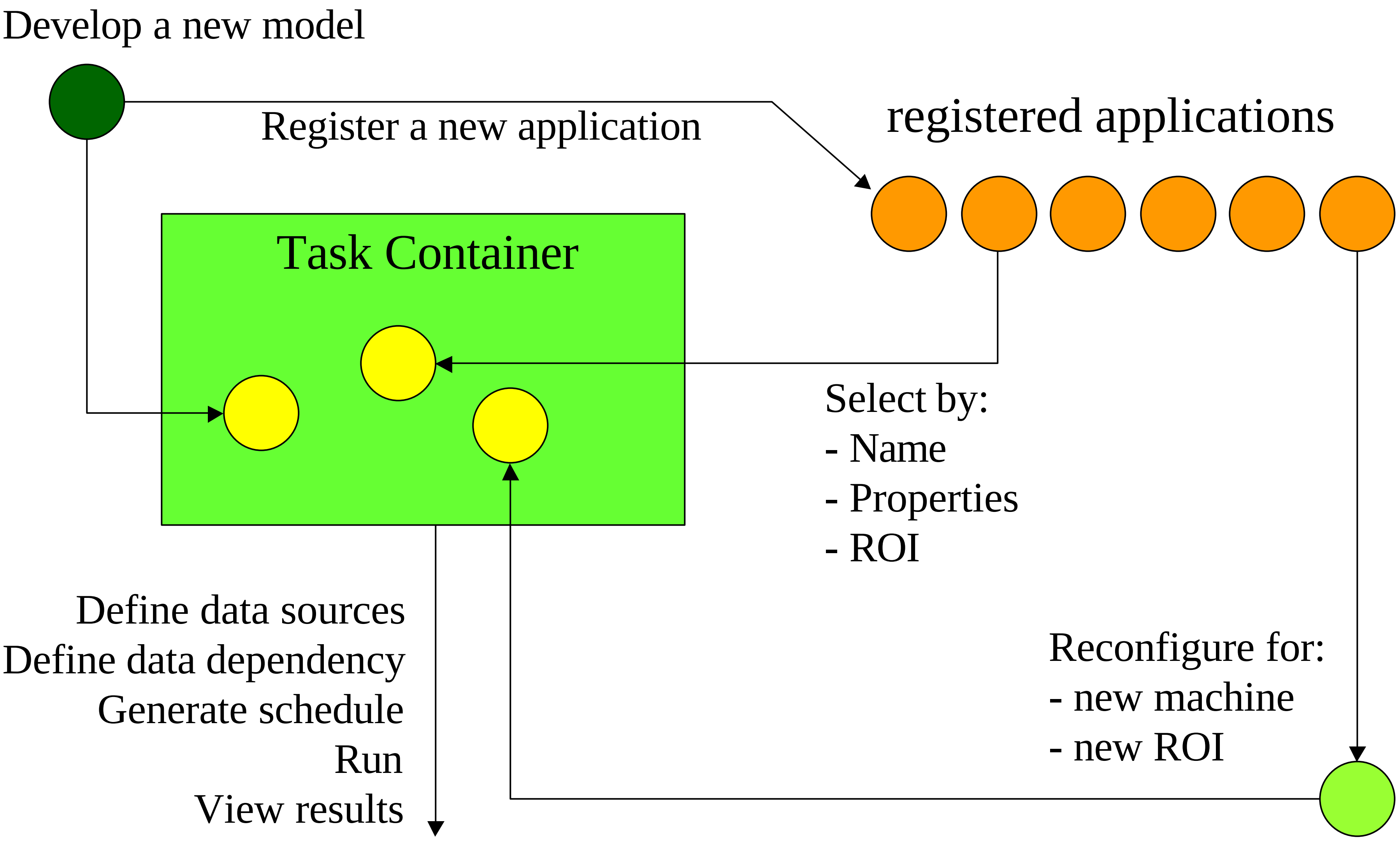
Private

- App2
- App3
- App4
- App5
- App6
- App12
- bbbbbgtt
- cvbnm
- dfghjk
- qwert
- Tom's application
- wamGui

© 2001 Mississippi State University

supermsparc 0 0
ultramsparc 0 3

Computational Task



Develop a new model

Register a new application

registered applications

Task Container

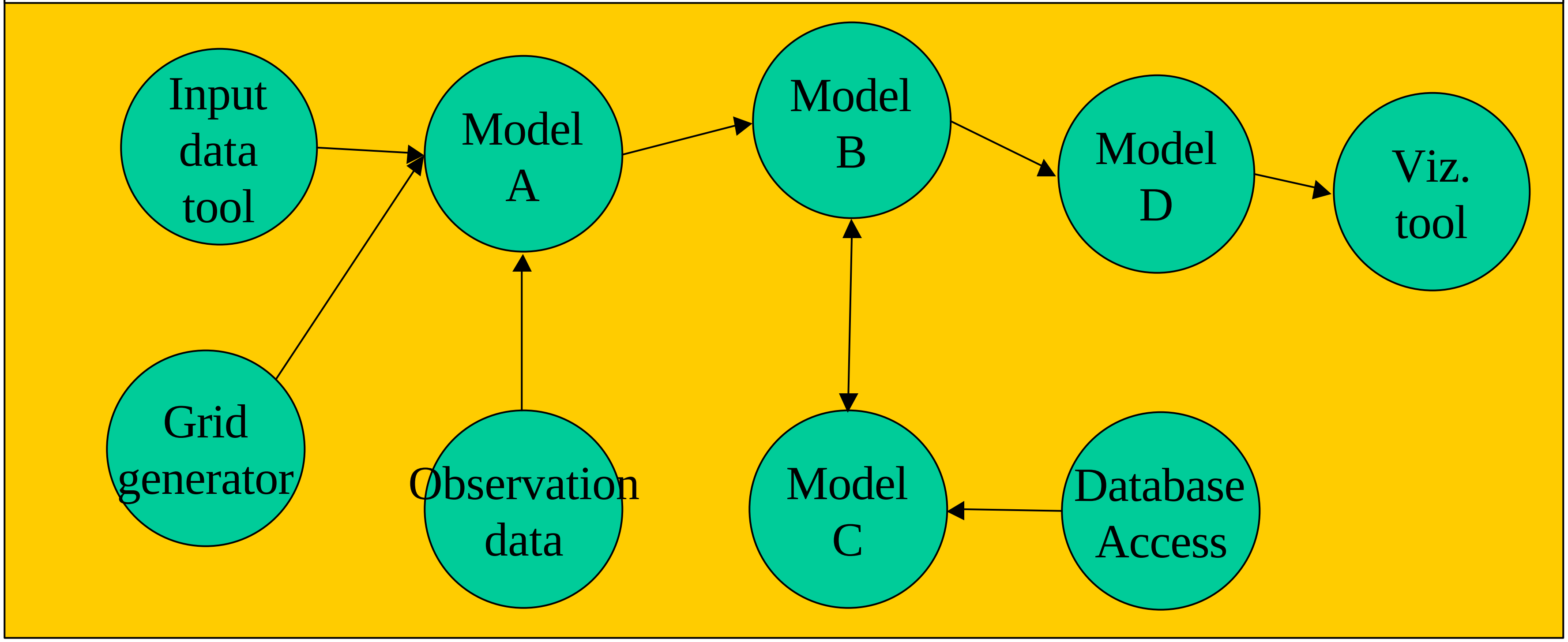
Select by:
- Name
- Properties
- ROI

Define data sources
Define data dependency
Generate schedule
Run
View results

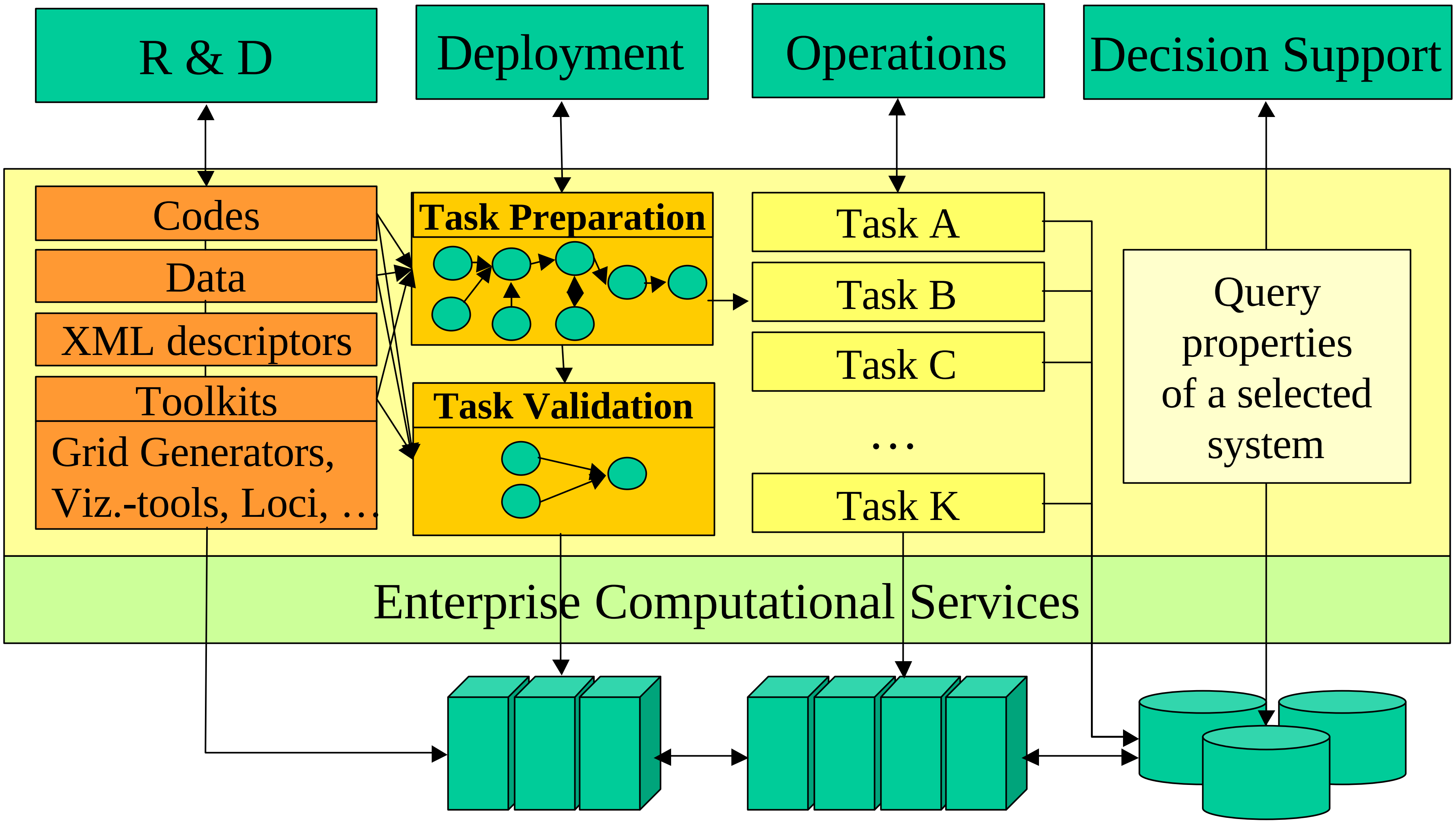
Reconfigure for:
- new machine
- new ROI

Task Preparation

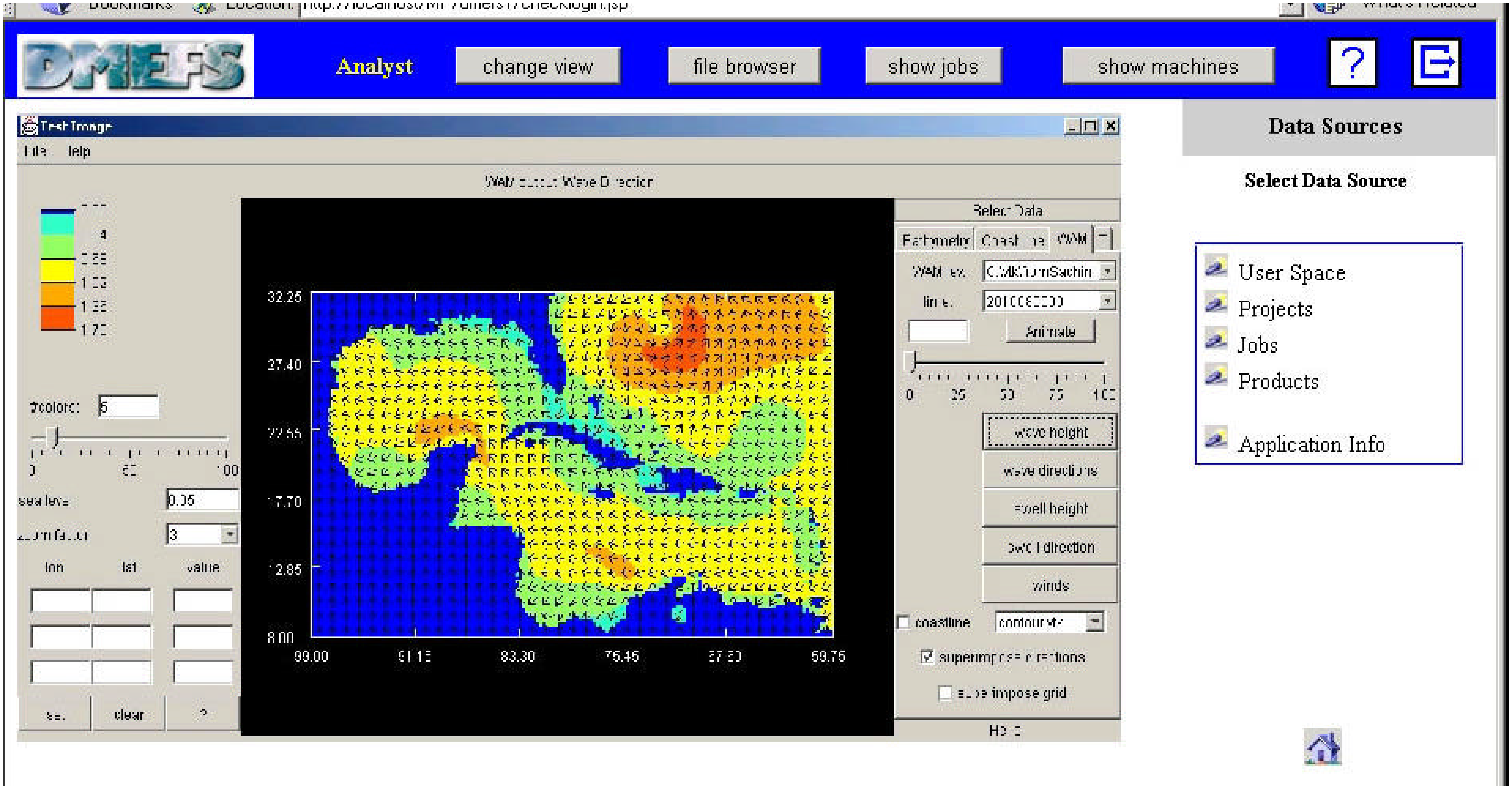
Task comprises: codes, data, tools and couplings
Couplings: the same address space, the same machine, geographically distributed, parameterized



DMEFS Functionality



Why we need persistence? (1)



WAM ex. C:\MK\TomSachin

time 2010080000

#colors: 5

sea level 0.05

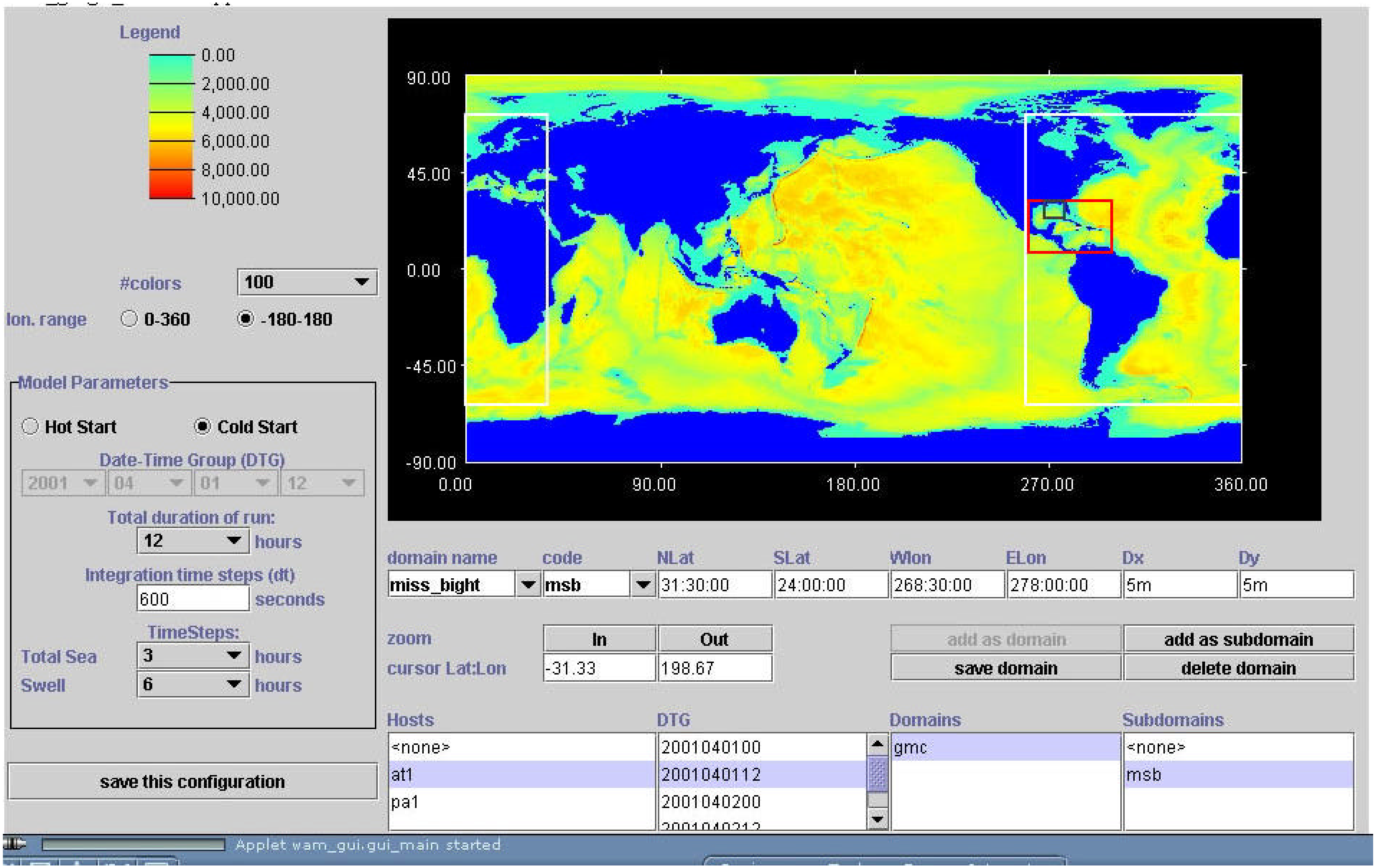
zoom factor 3

□ coastline contour

☑ superimpose directions

□ superimpose grid

Why we need persistence? (2)



#colors: 100

lon. range: ○ 0-360 ● -180-180

Model Parameters

○ Hot Start ● Cold Start

Date-Time Group (DTG): 2001 04 01 12

Total duration of run: 12 hours

Integration time steps (dt): 600 seconds

TimeSteps:

Total Sea: 3 hours

Swell: 6 hours

domain name	code	NLat	SLat	Wlon	ELon	Dx	Dy
miss_bight	msb	31:30:00	24:00:00	268:30:00	278:00:00	5m	5m

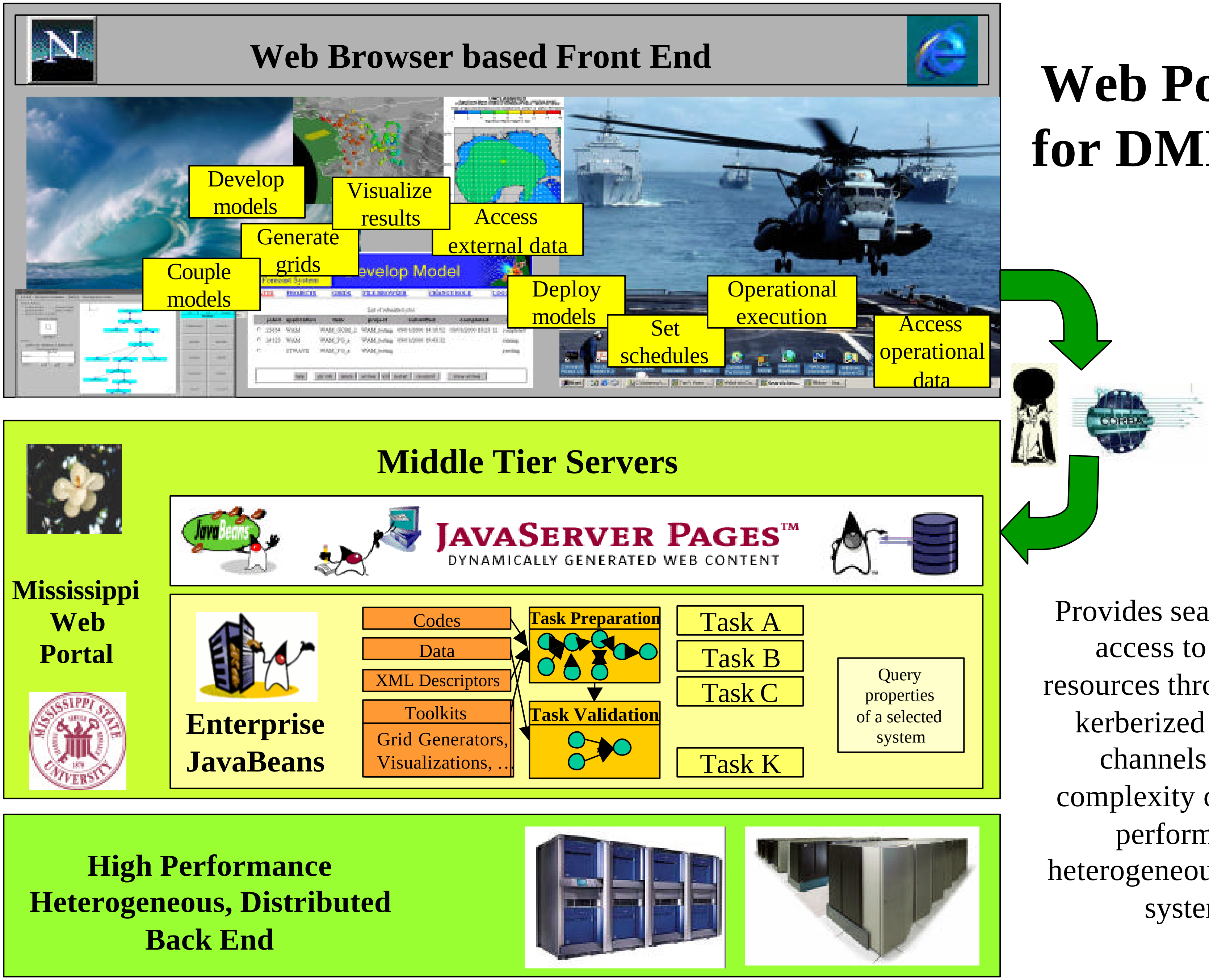
zoom

cursor Lat:Lon: -31.33 198.67

Hosts	DTG	Domains	Subdomains
<none>	2001040100	gmc	<none>
at1	2001040112		msb
pa1	2001040200		
	2001040212		

Applet wam_gui.gui_main started

Web Portal for DMEFS



Web Browser based Front End

Develop models

Visualize results

Access external data

Generate grids

Couple models

Deploy models

Operational execution

Set schedules

Access operational data

Middle Tier Servers

Mississippi Web Portal

JAVASERVER PAGES™
DYNAMICALLY GENERATED WEB CONTENT

Enterprise JavaBeans

Codes

Data

XML Descriptors

Toolkits

Grid Generators, Visualizations, ..

Task Preparation

Task Validation

Task A

Task B

Task C

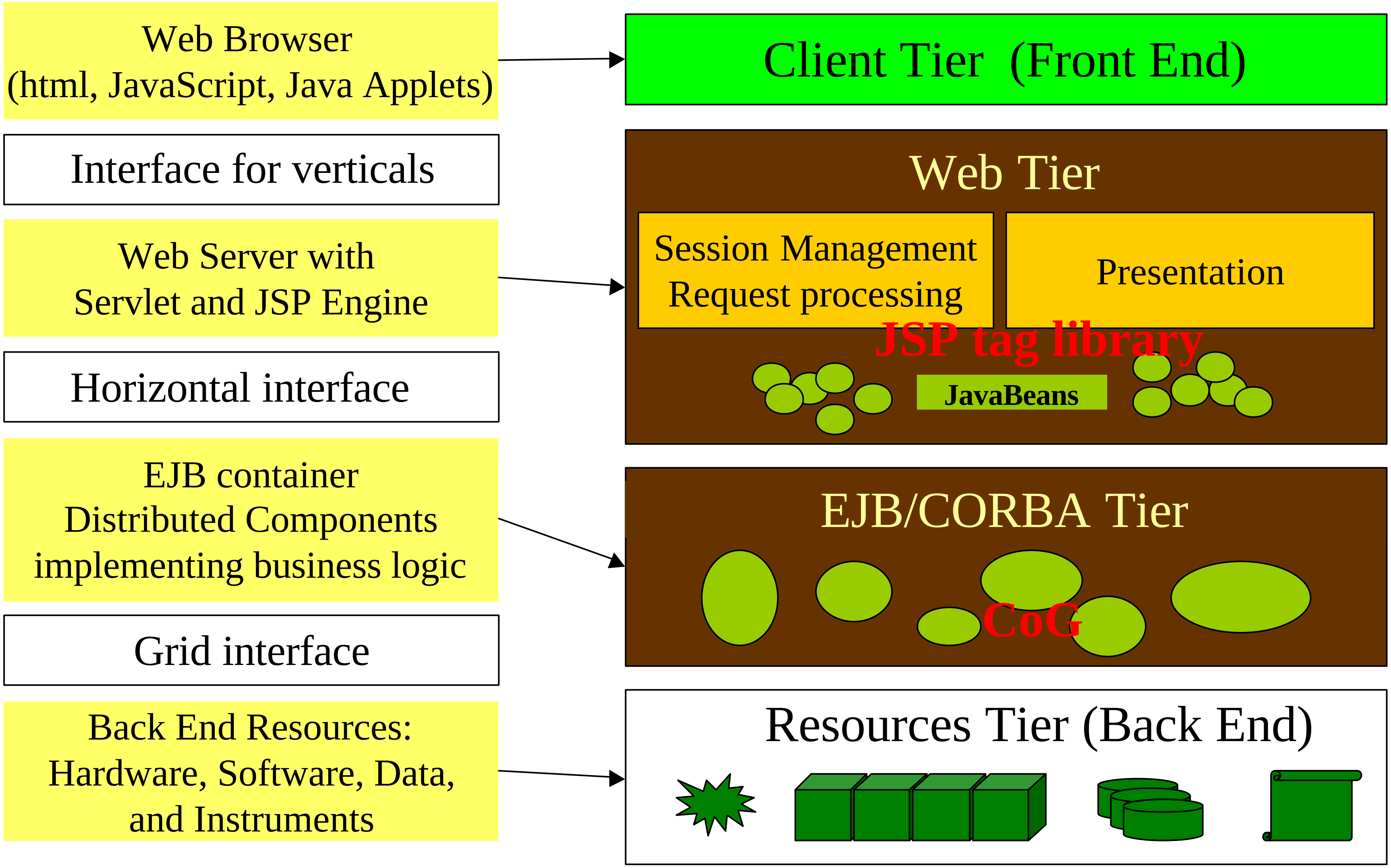
Task K

Query properties of a selected system

High Performance
Heterogeneous, Distributed
Back End

Provides seamless web access to remote resources through secure kerberized CORBA channels hiding complexity of the high performance, heterogeneous back-end systems.

Four-Tier Architecture of the Web Portal



Web Browser
(html, JavaScript, Java Applets)

Interface for verticals

Web Server with
Servlet and JSP Engine

Horizontal interface

EJB container
Distributed Components
implementing business logic

Grid interface

Back End Resources:
Hardware, Software, Data,
and Instruments

Client Tier (Front End)

Web Tier

Session Management
Request processing

Presentation

JSP tag library

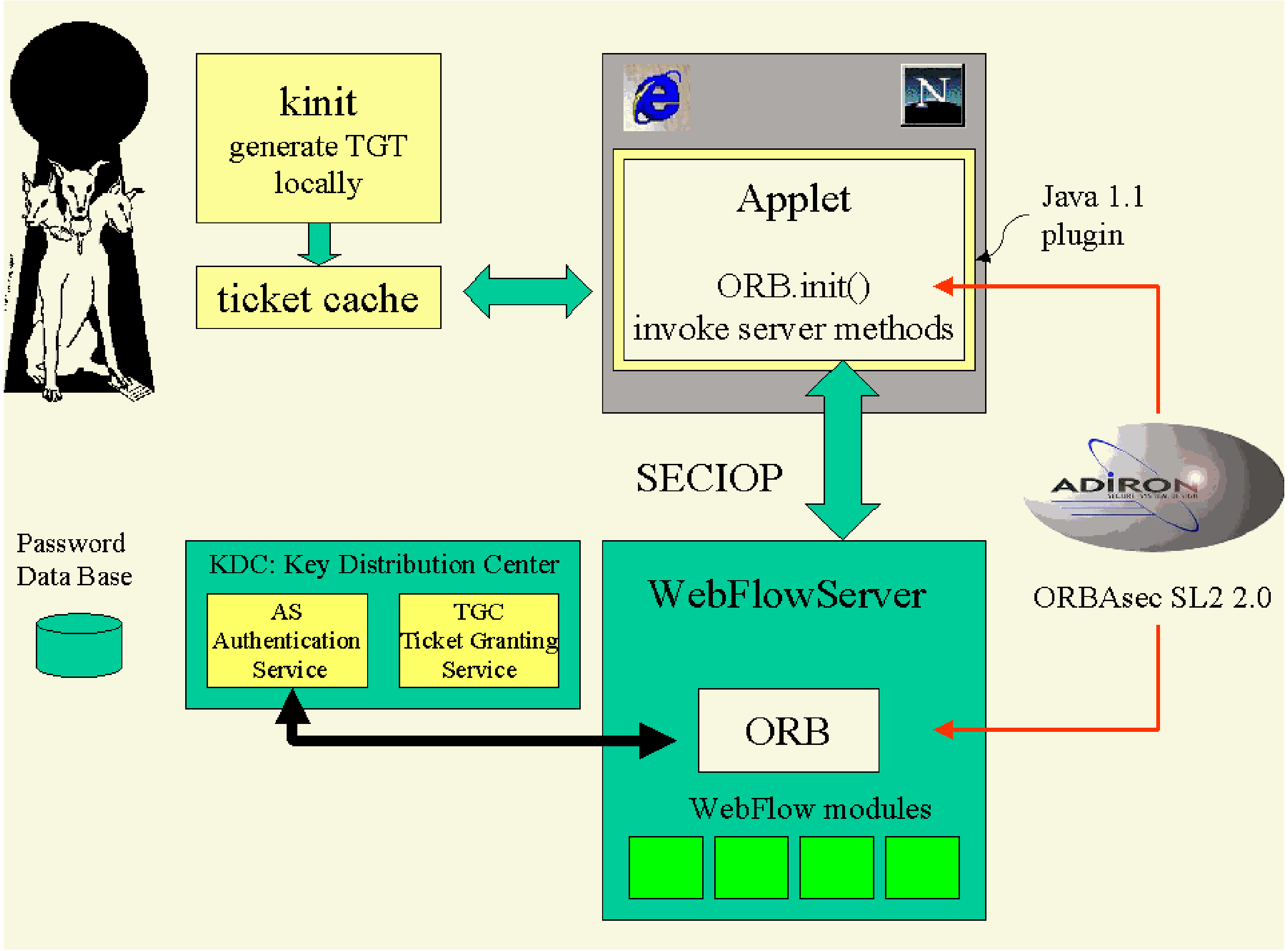
JavaBeans

EJB/CORBA Tier

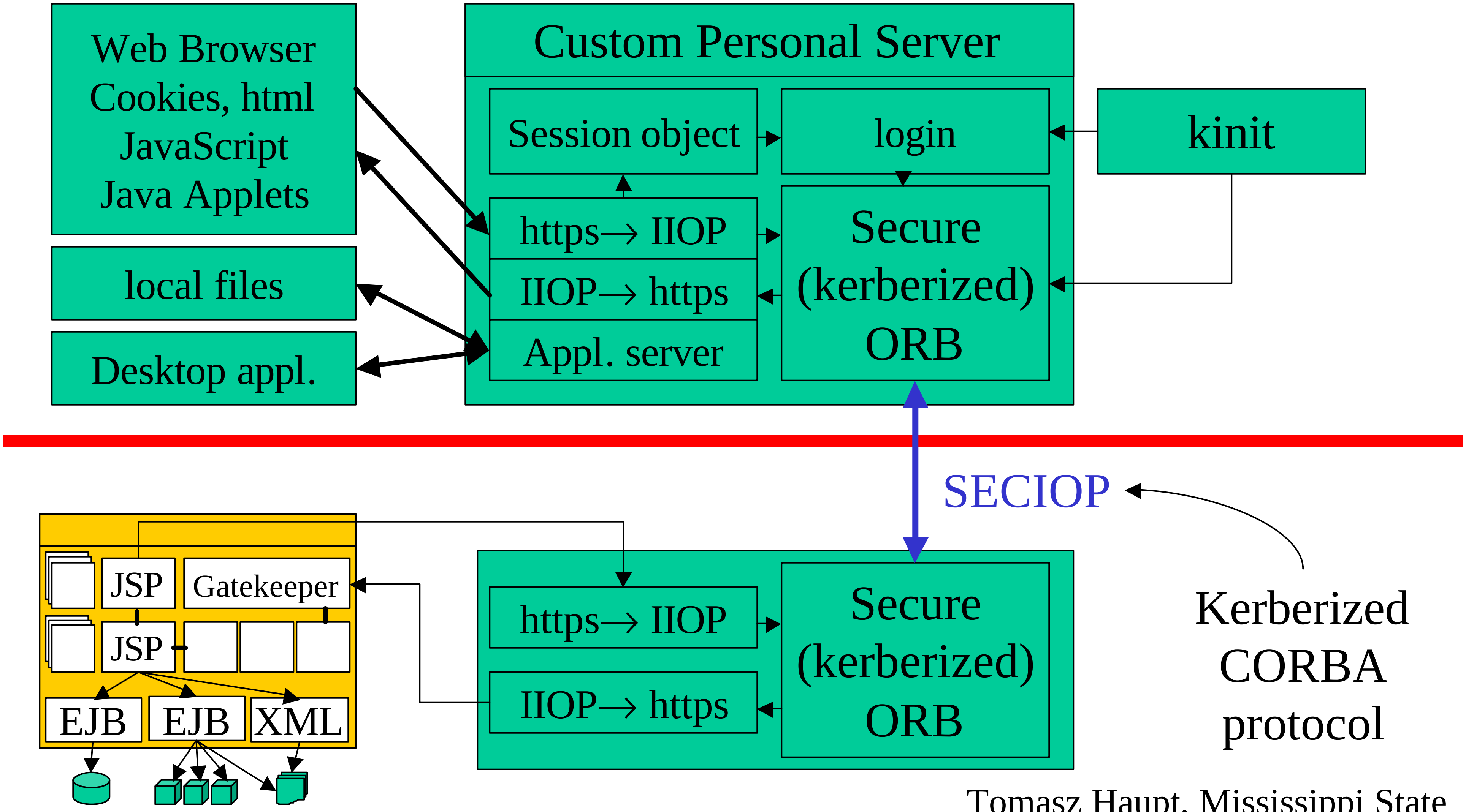
CoG

Resources Tier (Back End)

Security: CORBA security service



Details (Kerberized version)



Tomasz Haupt, Mississippi State

Summary

Features of Computational Web Portals

- extends the user desktop by providing a seamless access to remote resources.
- the user can state complex problems, allocate all resources needed to solve them, and analyze results.
- definitions of problems, methods of solving them, and their solutions are persistently stored; consequently they can be viewed and reused at later time, can be shared between researchers and engineers, and can be transitioned for operational or educational use.
- hides from the user complexity of heterogeneous, distributed, high performance back end.